The NVIDIA GeForce 8800 GPU

August 2007

Erik Lindholm / Stuart Oberman

Outline

- GeForce 8800 Architecture Overview
- Streaming Processor Array
  - Streaming Multiprocessor
  - Texture
- ROP: Raster Operation Pipeline
  - Coverage Sampled Anti-Aliasing
- The Big Picture

GeForce 8800 Architecture Overview

Graphics pipelines for last 20 years

Dedicated hardware per processing stage

Fixed Transform & Lighting, evolved to programmable vertex shading

Triangle, point, line – setup

Flat shading, texturing, evolved to programmable pixel shading

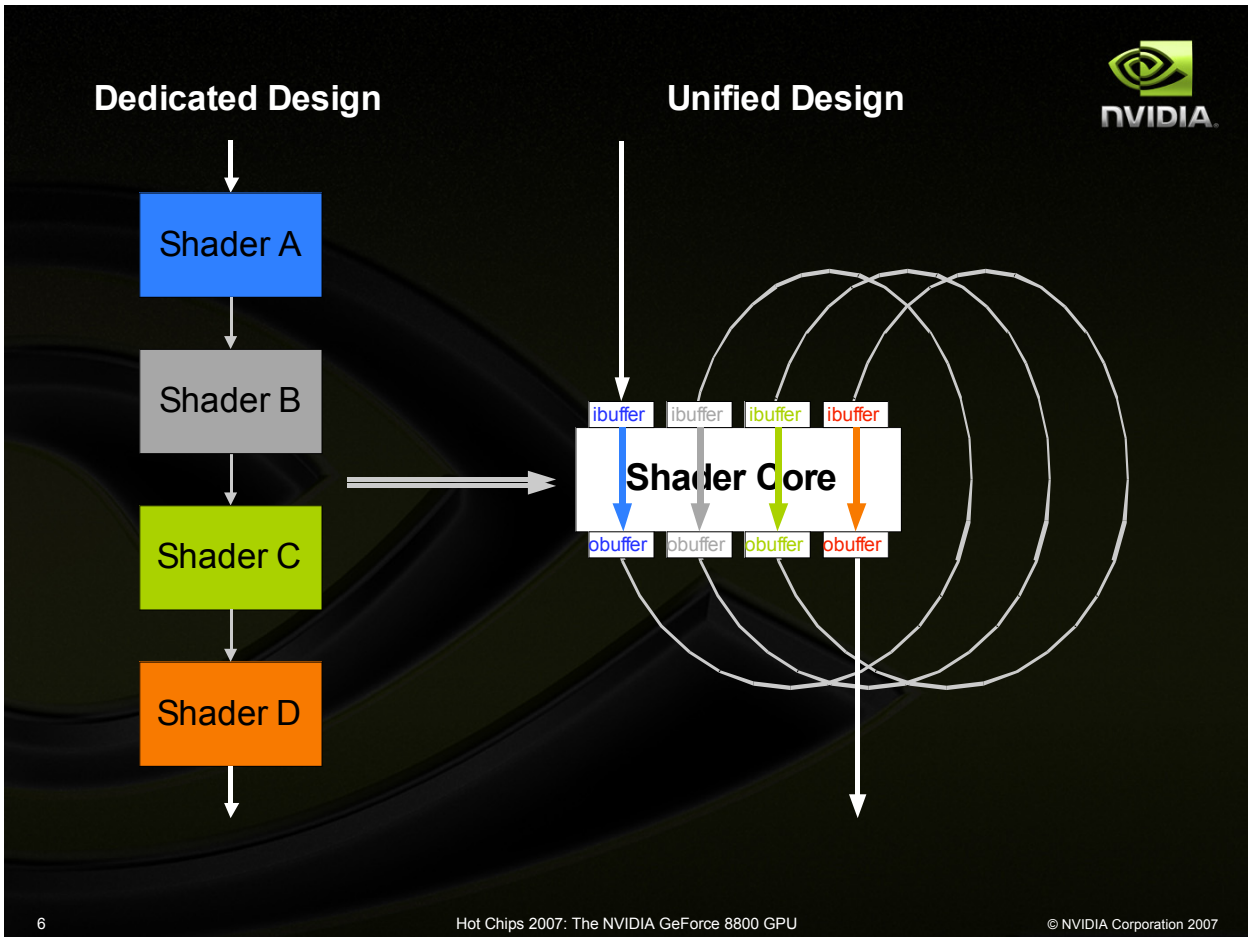
Blending, Z-buffering, antialiasing

Wider and faster over the years

CAGR DRAM bw: 1.4x raw, 2.0x with compression

Unified Shader Processor Architecture

- GeForce 8800 has a unified shader processor architecture
  - All shader stages use the same instruction set
  - All shader stages execute on the same units
    - vertex, geometry, pixel shaders
- Permits better sharing of expensive hardware shader resources
- Building dedicated units often results in under-utilization due to the workload of the application
  - When one unit becomes the bottleneck, other pipelined units are less efficient
- Dynamic and static thread/resource load balancing



March To Generality

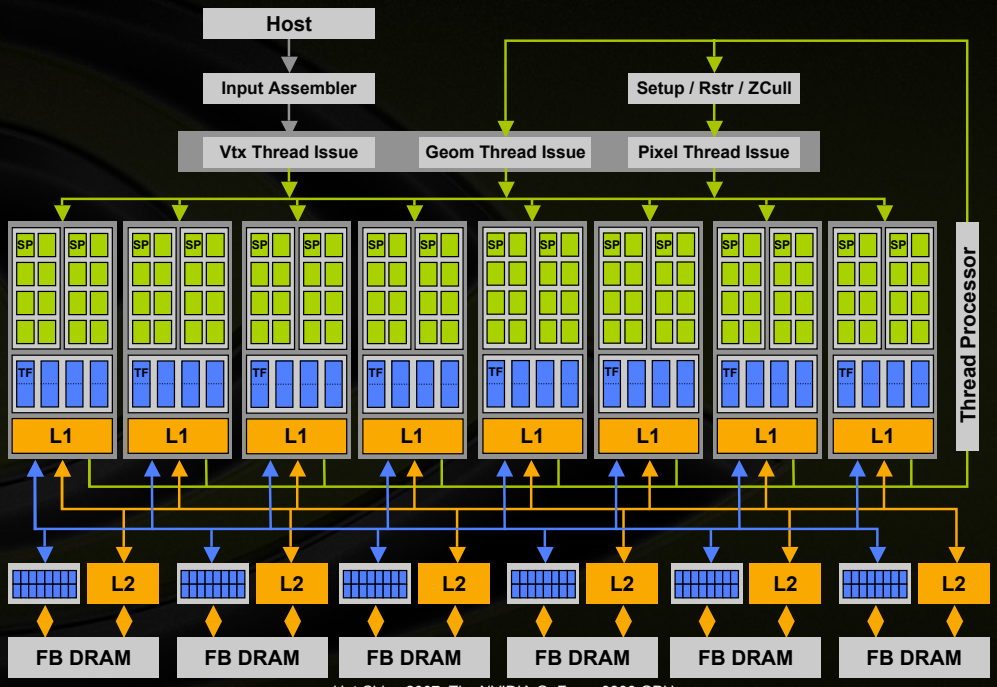
- Developers always want more flexibility
  - As a result, fixed-function shader units are becoming programmable
- Requires general programming model
  - More shader types
    - vertex, geometry, pixel, GPU compute
  - More instructions and instruction types
    - branching, integer, double precision floating point
  - More registers
  - More constants
  - More inputs/outputs
  - Load/store support
- Leads to GPU compute

Shader Model Progression

	DX8 SM1.x	DX9 SM2	DX9 SM3	DX10 SM4
Vertex Instructions	128	256	512	64k
Pixel Instructions	4+8	32+64	512	
Vertex Constants	96	256	256	16x4096
Pixel Constants	8	32	224	
Vertex Temps	16	16	16	4096
Pixel Temps	2	12	32	
Vertex Inputs	16	16	16	16
Pixel Inputs	4+2	8+2	10	32
Render Targets	1	4	4	8
Vertex Textures	n/a	n/a	4	128
Pixel Textures	8	16	16	
2D Tex Size			2k x 2k	8k x 8k
Int Ops	-	-	-	✔
Load Op	-	-	-	✔
Derivatives	-	-	✔	✔
Vertex Flow Control	n/a	Static	Static/Dyn	Dynamic
Pixel Flow Control	n/a	n/a	Static/Dyn	

GeForce 8800 Replaces the Pipeline Model

- The future of GPUs is programmable processing
- So – build the architecture around the processor



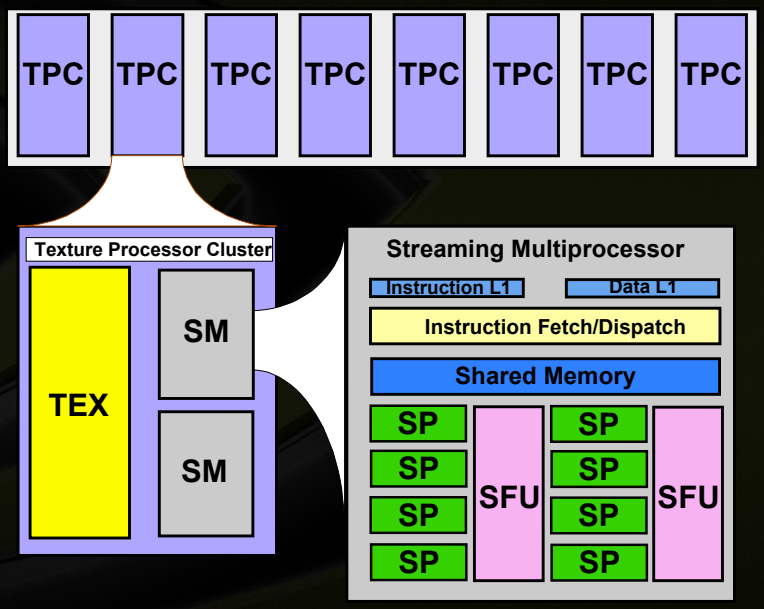
Streaming Processor Array

Streaming Processor Array

- SPA contains 8 Texture Processor Clusters (TPC)
- Each TPC contains 2 Streaming Multiprocessors (SM) and 1 texture pipe (TEX)
  - SM executes shader stages
- Communicates with the Raster Operation Processors (ROP) which have Frame Buffer (FB) memory access
- There are 6 ROPs and 6 64b FB partitions

Streaming Processor Array (SPA)

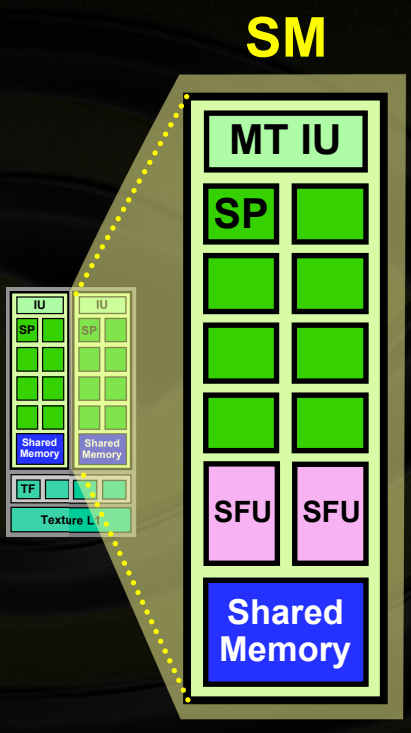
SPA = Streaming Processor Array (8 TPC)
TPC = Texture Processor Cluster (2 SM + TEX)
SM = Streaming Multiprocessor (8 SP + 2 SFU)
SP = Streaming Processor



Design Goals of the SM

- Independent processor and memory pipelines
  - Better memory latency hiding
  - Better for GPU Compute
- Unified Processor
  - In a fixed-function fully-pipelined architecture, a shader stage bottleneck stalls the entire pipeline
  - Unified design allows shader stage load-balancing
- Scalar ALUs instead of vector
  - Shader programs are becoming longer and more scalar, hard to be efficient with a vector architecture
- Compilable
  - Shader programs should be compiler-friendly

SM Multithreaded Multiprocessor



- SM has 8 SP Thread Processors
  - 36 GFLOPS peak at 1.50 GHz
  - IEEE 754 32-bit floating point
  - 32-bit integer
- SM has 2 Special Function Units
- Scalar ISA
  - Memory load/store
  - Texture fetch
  - Branch, call, return
- 768 Threads, hardware multithreaded
  - 24 SIMD warps of 32 threads
  - Independent MIMD thread execution
  - Hardware thread scheduling
- 16KB Shared Memory
  - Concurrent threads share data
  - Low latency load/store

SP Multiply-Add (MAD) Unit

- MAD unit operates on fp32 operands, produces fp32 output
- Performs all fundamental FP operations:
  - FADD, FMUL, FMAD, FMIN, FMAX
- Performs integer ops and conversions
- Fully-pipelined, but latency is not over-optimized at the expense of area
- FADD and FMUL IEEE 754 compliant
  - Round-to-nearest-even and round-to-zero
  - Special numbers properly handled
  - Denormal inputs and outputs are flushed-to-zero

Special Function Unit (SFU)

- Transcendental function evaluation and per-pixel attribute interpolation
- Function evaluator:
  - rcp, rsqrt, log2, exp2, sin, cos approximations
  - Uses quadratic interpolation based on Enhanced Minimax Approximation
  - 1 scalar result per cycle
- Attribute interpolator:
  - Evaluates attribute plane equations per pixel
  - 4 scalar results per cycle (pixel quad)

SFU: Attribute Interpolation

- Plane equation unit generates plane equation fp32 coefficients to represent all triangle attributes
- A, B, and C are fp32 interpolation parameters associated with a given triangle's attribute U
- Resulting attribute value U is fp32
- SFU must interpolate the value of each attribute per (x,y) for all pixels to be drawn:
  - U(x,y) = A*x + B*y + C
- For perspective correct interpolation:
  - Interpolate 1/w, and reciprocate to form w
  - Interpolate U/w
  - Multiply U/w and w to form perspective-correct U

Transcendental Function Statistics

Function	Input Interval	Accuracy (good bits)	Ulp error	% exactly rounded	Monotonic
1/X	[1,2)	24.02	0.98	87%	Yes
1/sqrt(X)	[1,4)	23.40	1.52	78%	Yes
2^X	[0,1)	22.51	1.41	74%	Yes
$\log_2 X$	[1,2)	22.57	n/a	n/a	Yes
Sin/cos	[0,pi/2)	22.47	n/a	n/a	No

Dynamic Load Balancing – Company of Heroes

Less Geometry

More Geometry

High pixel shader use

Low vertex shader use

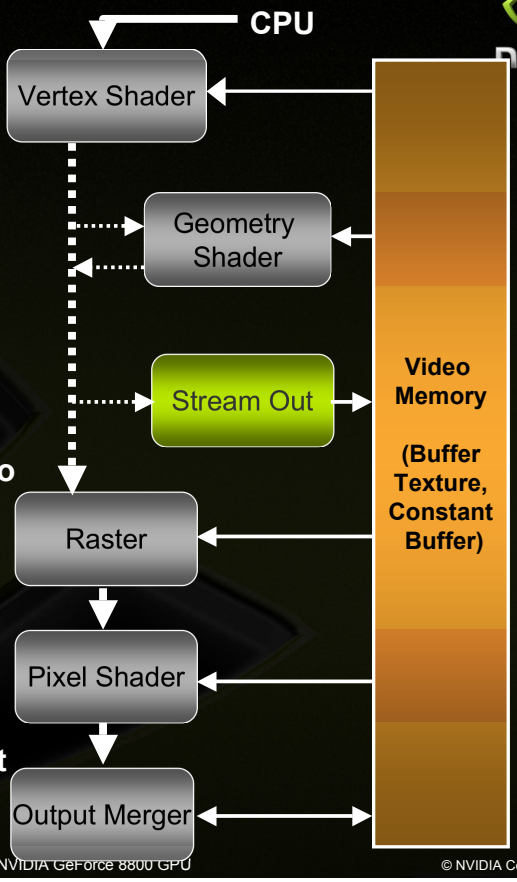
Balanced use of pixel shader and vertex shader

Unified Shader Usage

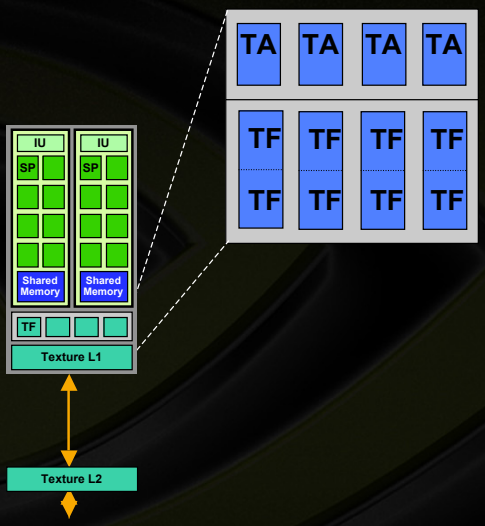
Company of Heroes image courtesy of Relic Entertainment

Geometry Shader and Stream Out

- Geometry shader sits between the vertex and setup/raster stages in the logical pipeline
- Processing of whole primitives
- Typical uses are stencil shadow polygon generation, skinning, and environment map creation
- Stream-out allows the output of vertex and geometry information to the frame buffer
- Data replay enables multi-pass operations such as recursive subdivision, storing skinning results, particle systems, and physics simulations
- Many of these operations were traditionally performed by the host CPU, and can now be moved into the graphics pipe

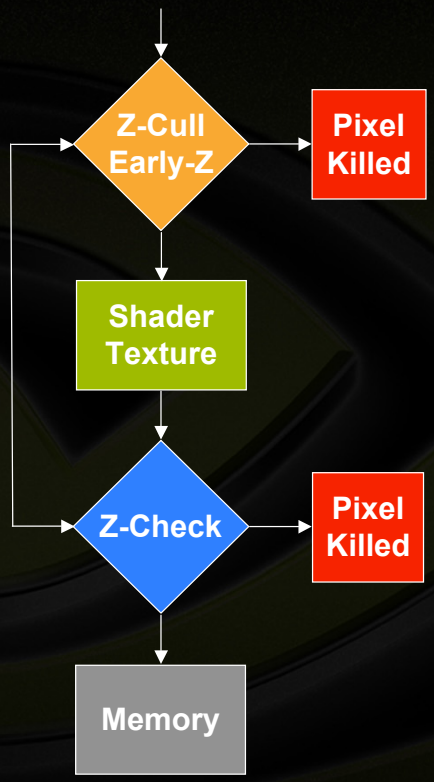


Texture



- 32 ppc Texture Address
- 64 ppc Texture Filter
- 38.4 GBilerps/sec (0.6 GHz)
- Optimized for HDR (High Dynamic Range) formats
- Support fp16 and fp32
- Full speed Aniso (2:1)
- Full speed fp16 HDR
- Highest image quality

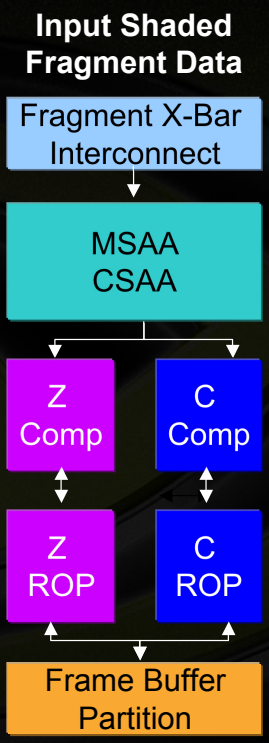
Z Cull and Early-Z



- To maximize efficiency, Z cull and early-Z allow non-visible pixels to be rejected after the raster subsystem and before expensive pixel shader processing
- Z cull rejects large blocks of pixels using coarse Z data at high rate
- Early-Z performs a second level of rejection at pixel granularity based on the contents of the actual Z buffer

ROP:
Raster Operation Pipeline

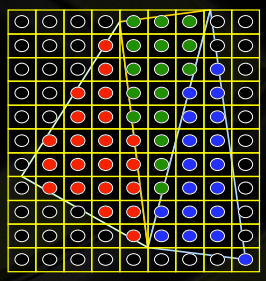
Detail of a single ROP pixel pipeline



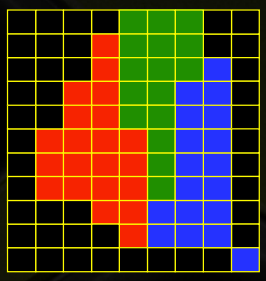
- FP16 & FP32 Support
- Up to 16x Multisampling
- AA with HDR
- 16 Samples + 16 Z Samples per partition
- Up to 32ppc Z-Only (per partition)
- New Color & Z Compression (2x)
- Multiple Render Targets (8)
- 96 Color Samples + 96 Z Samples per clock
- 192 samples Z-Only (when color writes are disabled)

Anti-Aliasing

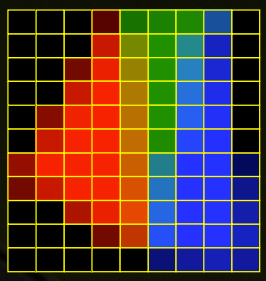
- Anti-aliasing properly accounts for the contribution of all the primitives that intersect a pixel



Triangle Geometry



Aliased



Anti-Aliased

Multisampling

- Store a unique color and depth value for each pixel sub-sample, but re-use one calculated color for all color sub-samples
- Strengths
  - Only one color value calculated per pixel per triangle
  - Z and stencil evaluated precisely; interpenetrations and bulkheads correct
- Weaknesses
  - Memory footprint N times larger than 1x
  - Expensive to extend to 8x quality and beyond
- Multisampling evolved from 1 → 2 → 4 samples
- Beyond 4 sub-samples, storage cost increases faster than the image quality improves
- Even more true with HDR
  - 64b and 128b per color sub-sample!
- For the vast majority of edge pixels, 2 colors are enough
  - What matters is more detailed coverage information

Coverage Sampled Antialiasing (CSAA)

- Compute and store boolean coverage at 16 sub-samples
- Compress the redundant color and depth/stencil information into the memory footprint and bandwidth of 4 or 8 multisamples

- Performance of 4xAA with 16x quality

- Low cost per coverage sample
- Just works with existing rendering techniques
  - HDR, stencil algorithms
- Efficient use of shader and texture hardware
- Boolean, not scalar, coverage
  - No bleed-through
- Fallback to the stored sample count quality (4x or 8x) for high-contrast Z/stencil results
  - Inter-penetrating triangles
  - Stencil shadow volumes

Antialiasing Modes Comparison

AA Mode:	Brute-Force Supersampling			Multisampling			Coverage Sampling		
Quality level:	1x	4x	16x	1x	4x	16x	1x	4x	16x
Texture/Shader Samples	1	4	16	1	1	1	1	1	1
Stored Color/Z Samples	1	4	16	1	4	16	1	4	4
Coverage Samples	1	4	16	1	4	16	1	4	16

Multisampling reduces texture & shader work

Coverage Sampling reduces color & Z footprint & bandwidth

The Big Picture

GeForce 8800 Scalability

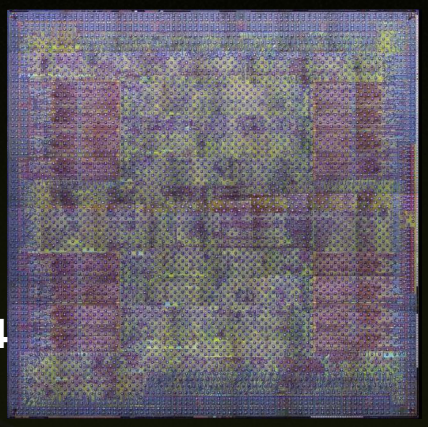
- Architecture was designed for scalability
  - value, mainstream, enthusiast segments
- Number of SMs, TPCs and ROPs can be varied allowing the right mix for different performance and cost targets
- Upward scalability is available with Scalable Link Interconnect (SLI), allowing multiple GPUs to be connected together

GeForce 8800 Performance

	7900GTX	8800GTX
Shader Model	SM3	SM4
Vertex Shader Units	8 (vector)	128 (scalar)
Pixel Shader Units	24 (vector)	
Shader Math (GFlops)	232	576
Texture Filter	12 GBilerp/sec	38 GBilerp/sec
ROP Processing	Up to 24ppc	Up to 192ppc
Memory Width	256-bit	384-bit
Memory Bandwidth	51.2 GB/sec	104 GB/sec

GeForce 8800 Implementation

- 681 million transistors, 470 mm^2
- Manufactured in TSMC 90nm
- Multiple clock domains
- 384 pin memory interface connecting to 768 MB of DDR frame buffer memory, yielding 104 GB/sec of bandwidth (1.08 GHz)
- Typical operating power consumption of 150 W



GeForce 8800 Summary

- Processor based architecture
  - previous architectures are graphics pipeline based
- Scalable number of processing cores
- 576 GFLOPS/sec just for shader execution (1.5 GHz)
  - scalar processors instead of 4-vector
- Hardware multithreading
  - 12288 threads
  - zero-overhead thread scheduling
- Scalable number of memory partitions
  - supports non-power of 2 partition count
- Adds GPU Compute